MINI PROJECT

Aim:

Write a code in JAVA for a simple Word Count application that counts the number of occurrences of each word in a given input set using the Hadoop Map-Reduce framework on local-stand alone set-up.

Solution :

Step 1: Open Eclipse >F|le>New> Java Project > (Give Name it - MR Programs Demo) > Finish

Step 2: Right Click>New>Package (Give Name it-Package n05)>Flnish

Step 3: Right Click on Package > New > Class(Give Name it - Word Count)

Step 4: Add Following Reference Libraries

Right Click on Project>Build Path>Add External Archivalst

/usr/lib/hadoop-0.20/hadoop-core.jar

Usr/lib/hadoop-0.20/lib/Commons-cli-1.2.jar

Step 5: Type following Program

```
packagePackage Demo5;
importjava.io.IOException;
importorg.apache.hadoop.conf.Configuration; importorg.apache.hadoop.fs.Path;
importorg.apache.hadoop.io.Int overline W ritablc
importorg.apache.hadoop.io.Long Writable;
importorg.apache.hadoop.io.Text;
importorg.apache.hadoop.mapreduceJob;
importorg.apache.hadoop-mapreduce Mapper;
importorg.apache.hadoop.mapreduce.Reducer;
importorg.apache.hadoop.mapreduce.lib.input.FileInput Format;
importorg.apache.hadoop.mapreduce.lib.output.FileOutput Format:
importorg.apache.hadoop.util.Generic OptionsParser; publicclass WordCount(
publicstaticvoidmain(String[]args) throws Exception
{
Configurationc=newConfiguration();
String files = newGenericOptionsParser(c,args).getRemainingArgs();
Pathinput = newPath(files[0]);
Pathoutput = newPath(files[1]);
Jobj=newJob(c,"wordcount");
```

```
J.setJarByClass(WordCount class);

j.setMapperClass(MapFor WordCount.class);

j.setReducerClass(Reduce For WordCount.class);

j.setOutputKeyClass(Text.class);

j.setOutputValueClass(IntWritable.class);

FileInputFormat.addInputPath(j,input);

FileOutput Format.setOutputPath(j,output):

System.exit(j.waitForCompletion(true)?0:1) ;

}

publicstaticclassMapForWordCountextendsMapper<LongWritable,Text, Text, IntWritable>{publicvoidmap(Long Writablekey,
Textvalue,Contextcon) throwslOException. InterruptedException

{

Stringline value.toString();

String[]words=line.split(",");

for(Stringword:words)

{

TextoutputKey =newText(word.toUpperCase().trim());

IntWritableoutput Value=newint Writable(1);

con.write(outpatkey.outputValue);

}

}

}

publicstaticclassReduceForWordCountextendsReducer <Text IntWritable, Text, IntWritable>

{

publicvoidreduce(Textword.Iterable<IntWritable>values.Contextcon)throwslOException, InterruptedException

{

intsum=0;

for(IntWritablevalue:values):

{

sum+value.get():

}

con.write(word.newIntWritable(sum));

}

}
```

}

Explanation:

- The programconsistsof3classes:
- Driverclass (Public void static main-the entry point)
- Mapclass which extends public class Mapper <KEYIN, VALUEIN, KEYOUT, VALUEOUT> and implements the Map function. Reduce class which extends public class Reducer <KEYIN, VALUEIN, KEYOUT, VALUEOUT> and implements the
- Reduce class with extend public class reducer < KEYIN ,VALUEN ,KEYOUT, VALUEOUT > and implements the Reduce function.

Step 6: Make JarFile

- Right Click on Project > Export > Select export destination as JarFile> next> Finish

Step 7: get at extfile and shift it in HDFS

- To Move this file into Hadoop directly, open the terminal and enter the following commands:

[training@localhost-Shadoopfs-putwordcountFilewordCountFile]

Step 8: Execute Jarfile

(hadoopjarjarfilename.jarpackageName.ClassNamePathTolnputTextFilePathToOutput Directry)

[training@localhost-Shadoopjar MRPrograms Demo jur Package Demo, Word

CountwordCountFileMRDir]

Step 9: OpenResult

[training@localhost-$hadoopfs-IsMRDicl

Found3items

-rw-r-r-trainingsupergroup02018-05-2003:36/user/training/MRDirl/_SUCCESS

drwxr-xr-x-trainingsupergroup02018-05-2003:36/user/training/MRDirl/_logs

-rw-r--Itrainingsupergroup202018-05-2003:36/user/training/MRDirl/part-r-00000

[training@localhost-Shadoopfs-catMRDirl/part-1-0000

BUS 7

CAR 4

TRAIN 6